Project Thesis (2013-14)

Submitted to **NIT Rourkela** in partial fulfilment of requirements for the **Award** of **B-Tech Degree** during Academic Year 2013-2014.

By

Manoranjan Minz (110EC0172)

Under the Esteemed Guidance of

Prof. Sukadev Meher,
HOD,
Department of ECE,
NIT Rourkela.

Efficient Image Compression Scheme for Still Images

An undertaking by

Manoranjan Minz
B-Tech Student,
Department of Electronics & Communication Engineering,
National Institute of Technology, Rourkela.

Project Guide:
Prof. Sukadev Meher,
HOD,
Department of ECE,
NIT Rourkela.

ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA
ROURKELA – 769008; www.nitrkl.ac.in

CERTIFICATE

This is to certify that the thesis titled "**Efficient Image Compression Scheme for Still Images**" submitted by Sri **Manoranjan Minz** in partial fulfillment of **Bachelor of Technology** in **Electronics and Communication Engineerin**g at **National Institute of Technology, Rourkela** is an authentic work carried out by him under my supervision and guidance.

Prof Sukadev Meher
Department of Electronics and
Communication Engineering
National Institute of Technology, Rourkela

DECLARATION

I, hereby declare that the project work entitled "**Efficient Image Compression Scheme for Still Images**" is a record of our original work done under **Prof. Sukadev Meher** in **National Institute of Technology, Rourkela**. Throughout this documentation wherever contributions of others are involved, every endeavor was made to acknowledge this clearly with due reference to literature. This work is being submitted in the partial fulfillment of the requirements for the degree of **Bachelor of Technology in Electronics and Communication Engineering** at National Institute of Technology, Rourkela for the academic session 2010–2014.

……

Manoranajan Minz,
B-tech Student,
Department of Electronics & Communication Engineering,
National Institute of Technology, Rourkela.

ACKNOWLEDGEMENT

An endeavour over a long period can be successful only with the advice and support of many well-wishers. I now take this opportunity to express our gratitude and appreciation to all of them.

I would like to express my gratitude towards my **Parents** for their kind co-operation and encouragement which help me in completion of my final year project.

I express my most heartfelt gratitude to my guide **Prof, Sukadev Meher, HOD**, Department of ECE, NIT Rourkela, for his invaluable guidance, technical supervision, valuable feedback and constructive discussions & suggestions for improving the quality of my project at NIT Rourkela. Without his involvement and supervision it would have been impossible to complete final year project.

Last but not the least; I would like to express my special gratitude and thanks to my friends, especially **Subhajit Sahu** for giving me such attention and time for my project.

.....

Manoranajan Minz,
B-tech Student,
Department of Electronics & Communication Engineering,
National Institute of Technology, Rourkela.

CONTENTS

SECTION II

ABSTRACT

The raw image files take a large amount of disk space and it can be a huge disadvantage while storing or transmitting, and for this reason an efficient image compression technique is required. Image compression has been a widely addressed research area since many years. Many compression standards have been proposed. But there is still a scope for high compression with better quality reconstruction. Discrete Cosine Transform (DCT) is used for compression in a very popular standard called JPEG. First section of this paper aims at the analysis of compression using DCT transform method, and results obtained from certain experiments and execution of programs in MATLAB have been shown. There are many compression techniques, but still a technique which is better, simple to implement and with a good efficiency is suitable for user requirements. In the second section of this paper a new lossless variable length coding method is proposed for image compression and decompression which is inspired by a popular coding technique called Huffman coding. This new variable length coding technique is simple in implementation and the resulting compressed file utilizes comparatively less memory than the actual image. By using the new variable length coding technique a software algorithm has been written and implemented for compression and decompression of an image in a MATLAB platform.

SECTION I

1. INTRODUCTION

Image compression algorithms are used to re-encode the original image data into more compact representations so that it will take less space on the disk. It can be thought of as using fewer words to say the same thing without losing its meaning. Ideally, an image compression technique removes redundant information, and encodes the remaining information efficiently. But in practical usage, it is often necessary to remove both non-redundant information and relevant information to achieve the required compression.

There are two types of compression techniques: one is Lossy compression method and another is Lossless compression method. Lossy compression creates a smaller file by removing (discarding) some information (which are less important) from the original image. It removes some details like variation in color, variations which are so minimal that the human eye cannot detect (visually lossless). Whereas in lossless compression no information is discarded or removed, the image which is reconstructed from the compressed image file is completely equal to the original image.

1.1.1 Lossy compression methods:

- Chroma subsampling technique
- Fractal compression technique
- Transform coding (DFT/DCT/ Wavelet transform)

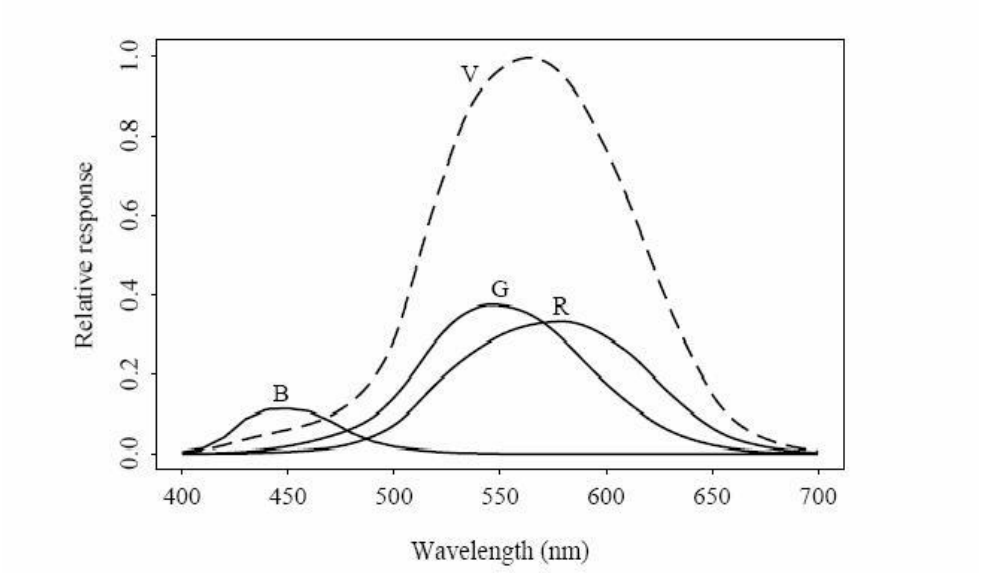
1.1.2 Lossless compression methods:

- Run length Coding (used in PCX, BMP, TGA, TIFF)
- DPCM (Differential pulse code modulation) and Predictive Coding

- Entropy Coding technique
- Adaptive dictionary algorithms like LZW (used in TIFF and GIF)
- Chain codes

1.1.3 Human Visual System:

The light focused by the lens on retina is sensed by Rods and Cones. Sensation of color is provided by cones and sensation of light intensity is provided by rods. There are three types of cones and each corresponds to red, green and blue colors. The sensitivity of eye towards various wavelengths is shown in figure 1.1.



(Figure 1.1)

The sensitivity of cones towards different wave lengths is shown by curves R, G and B. The response of eye intensity variation is shown by curve V. It's clear from the figure 1.1 that the human eye is more sensitive towards light intensity variation and less sensitive towards the color variation. A lossy compression technique called color sub-sampling (Chroma sub-sampling) uses this principle.

1.2 Digital Domain Representation of images:

An image consists of small elements called pixels. It has some finite numbers of row and column pixels. Some of the properties like this of digital images are discussed below.

1.2.1 Pixel

Any color can be derived by combining the primary colors Red, Green and Blue in a proper ratio. Magenta, yellow, and cyan are the secondary colors and they can also be formed from the three primary colors. While representing an analog image, the intensity values of red, green and blue are required for each and every point in the space. But in Digital domain, the space is divided into a set of small finite square boxes. These finest square areas are called pixels. Each Pixel contains the intensity values of Red, Green and Blue in that square area. These Red, Green and Blue intensity components are separated by a Baye's Filter arrangement. Image sensors like CCD (Charge coupled devices), CID (Charge injection Devices), and active pixel CMOS are responsible for the finite division of space at which lens is focused. For grayscale digital images the value at each position can be approximated to any of 256 intensity values if 8 bits/pixel is used for intensity value representation. In color images 8 bits are needed for representing each of the primary colors, so 24 bits/pixel is required. 24 bits/pixel means, there can be 2^24 = 16,777,216 color levels for each pixel in the space. The number of bits/pixel is called color depth, which is a measure of Radiometric resolution.

1.2.2 Resolution

It's a measure of quality of an image. There are many different types of resolutions in digital images, such as Spatial resolution, Pixel resolution, Spectral resolution, Temporal resolution and Radiometric resolution. But generally resolution is said to be the total numbers of pixels present in an image i.e. Pixel resolution. More the number of pixels the more will be the image clarity. To get a good quality image there should be at least 300pixels/inch.

1.2.3 RAW Formats

When there is no compression done on the image, the image format is said to be RAW format. There are three major RAW formats such as RGB, YUV and YIQ. It's explained in section 1.1.3 that human eye is more sensitive towards light intensity variation and less sensitive towards color variation. So the overall image quality will not be much affected, if there's some loss in color information. Information which is obtained from the image sensor is generally in RGB format. And to display the image on a device the same RGB format is required. For Analog TV transmissions such as NTSC and PAL, YUV and YIQ formats are developed respectively. And the digital formats like YUV, YCbCr are most commonly used in video compression and image compression.

Formulae for converting one format to other:

RGB --> YCbCr

$Y = 0.299R + 0.587G + 0.11B$

$Cb = 0.564(B - Y)$

$Cr = 0.713(R - Y)$

YCbCr --> RGB

$R = Y + 1.402Cr$

$G = Y – 0.344Cb – 0.714C r$

$B = Y + 1.722Cb$

Where,

Y = Value of Luminance

Cb = Value of Chrominance

Cr = Color difference

R = Red intensity value

G = Green intensity value

B = Blue intensity value

1.3 Need For Compression:

Considering an image of resolution 320 × 240, its size in the RAW format (if each of the primary color is of 8bits i.e. for each pixel it needs 24 bits) would be 320 ×240× 3 bytes = 230,400 bytes. When it comes to RAW video streaming of length 1sec and with a frame rate of 25frames per second, it needs 320 × 240 ×3 × 25 = 5760000 bytes (5.5 MB) of storage. And for a 80 minutes movie it would take 80×60×5.5 = 25.74 GB. That's a huge amount of storage for just a 320×240 resolution movie. That is why, need for image compression is felt, so that more number of files can be stored on the disk.

2.1 DCT (Discrete Cosine Transform):

A discrete cosine transform (DCT) can represent a stream of input points in terms of a sum of cosine functions having different frequencies. The JPEG process which is a popular lossy image compression method uses Discrete Cosine Transform. DCT and Fourier transform converts images from spatial domain to frequency domain and time domain to frequency domain respectively, to de-correlate the neighbouring pixels. DCT is a reversible transformation. Compression actually happens during the quatization stage, where the less important frequencies (high frequencies) are discarded; hence it's called lossy compression. To retrieve the image from the compressed file, in the decompression process, only the remaining important frequencies (low frequencies) are used. But due to deletion of information that was contained in high frequencies, the reconstructed image tend to have some degradation.

DCT is used get de-correlation between the image pixels. The transform co-efficients can be encoded after de-correlation without hampering compression efficiency.

2.1.1 1D DCT:

The DCT transformation of a one dimensional sequence having N elements is defined as

$$C(u) = \alpha(u)\sum_{x=0}^{N-1} f(x)\cos\left[\frac{\pi(2x+1)u}{2N}\right]$$

Where u = 0, 1, 2, 3…..up to N-1

And formula for inverse transformation is

$$f(x) = \sum_{u=0}^{N-1}\alpha(u)C(u)\cos\left[\frac{\pi(2x+1)u}{2N}\right]$$

Where x=0,1,2,3…. N-1

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & for \quad u = 0 \\ \sqrt{\frac{2}{N}} & for \quad u \neq 0. \end{cases}$$

And

For $u = 0,\ C(u=0) = \sqrt{\frac{1}{N}}\sum_{x=0}^{N-1} f(x)$

It can be observed that the first transform co-efficient is the mean value of the whole sample sequence. It's called as the DC co-efficient and the rest transform co-efficients are called AC co-efficients.

2.1.2 2D DCT:

The 2D DCT can be obtained by applying one dimensional DCT twice on the given sequence (2D array), once in the x-direction and once again in the y-direction.

The following DCT equation gives the (i,j)th element of the DCT transformation of an image.

$$D(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} p(x,y) \cos\left[\frac{(2x+1)i\pi}{2N} \right] \cos\left[\frac{(2y+1)j\pi}{2N} \right]$$

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } u = 0 \\ 1 & \text{if } u > 0 \end{cases}$$

Where,

p(x,y) = (x, y)th element of the image (p matrix)

N = size of the block on which DCT is applied

The above equation computes one element (i,j)th of the transformed image from the grayvalue of the pixel of the original image matrix (p). The JPEG compression uses a standard 8*8 block, that means N is equal to 8 and x & y range from 0 to 7. Thus the elements of DCT matrix D(i,j) would be equal to

$$D(i,j) = \frac{1}{4} C(i)C(j) \sum_{x=0}^{7} \sum_{y=0}^{7} p(x,y) \cos\left[\frac{(2x+1)i\pi}{16} \right] \cos\left[\frac{(2y+1)j\pi}{16} \right]$$

As cosine functions are used in DCT transformation, the DCT matrix which is obtained depends upon both horizontal and vertical frequencies. There is a simple way to get the DCT matrix, by using another matrix T.

$$T_{ij} = \begin{cases} \frac{1}{\sqrt{N}} & \text{if } i = 0 \\ \sqrt{\frac{2}{N}} \cos\left[\frac{(2j+1)i\pi}{2N}\right] & \text{if } i > 0 \end{cases}$$

An orthonormal set of elements are formed on the columns of T, that implies T is an orthogonal matrix and $T^{-1} = T^T$. While computing inverse DCT, T's orthogonality property will make the computation easier, since inverse of T would be just the transpose of T.

If M = Original image matrix

DCT of M can be calculated as

DCT(M) = D = T * M * T^T

Firstly matrix T is multiplied with the original image matrix M to transform the rows, and then again by multiplying T^T to TM, columns are transformed.

The first co-efficient of the resulting matrix corresponds to the low frequencies of the image. DCT co-efficients correlate to higher and higher frequencies of the image block while moving away in all direction from the first element, and the last element i.e. the most bottom-right element corresponds to highest frequency. The human eye is mostly sensitive to lower frequencies and less sensitive to high frequencies, so elements corresponding to high frequencies are the redundant elements and can be discarded.

2.2 Quantization:

Once the transformed matrix is calculated from the DCT transformation, it's needed to discard/filter out the redundancies i.e. high frequencies components which are not necessary.

For that purpose quantization is carried out by taking the help of quantization matrix of the same size as that of the image block or sub-image block.

The original image is divided in to 8*8 pixel blocks of sub-images in the JPEG standard. And to each of these 8*8 blocks DCT transformation is applied. Quantization matrices helps to select any quality level from 1 to 100, where Q1 quantization matrix gives the worst image quality and best compression, and Q100 gives the highest quality with lowest compression.
By dividing each element in the transformed matrix by the corresponding element in the quantization matrix, a Quantized matrix is formed, and then the outcomes are rounded to their nearest integer values.

C_{ij}= Round (D_{ij}/Q_{ij})

There's one standard quantization matrix denoted by Q50 whose elements are pre-defined. A quality level of 50 gives good compression ratio and also excellent image quality for the decompressed image.
Desired quality levels can also be obtained from the standard Q50 matrix by the following formulae.

Q(>50) = {(100-qualitylevel)/50} * Q50

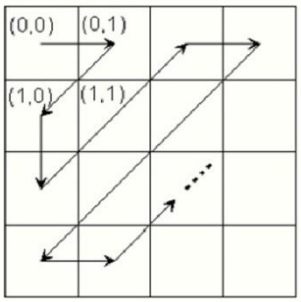
And Q(<50)= (50/quality level) * Q50

$$Q_{50} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

The quantized matrix is then rounded to the nearest positive integer values ranging from 1 to 255.

2.3 Coding:

Before storing the quantized matrix in memory or before transmission, all coefficients of the quantized matrix are converted to a bit-stream by an encoder. After applying quantization, many of the co-efficients may get equal to zero. JPEG encodes these quantized co-efficients in a zig-zag sequence and takes advantage of having large number of zeros.



As a result file gets compressed quite efficiently by combining consecutive zeros into one zero.

2.4 Decompression:

The quantized matrix can again be reconstructed by decoding the stream of bits. After that elements of the quantized matrix are multiplied with corresponding elements of the quantization matrix to obtain another matrix R.

$\mathbf{R_{ij} = Q_{ij} * C_{ij}}$

$\mathbf{Q_{ij}}$ is the Quantization matrix

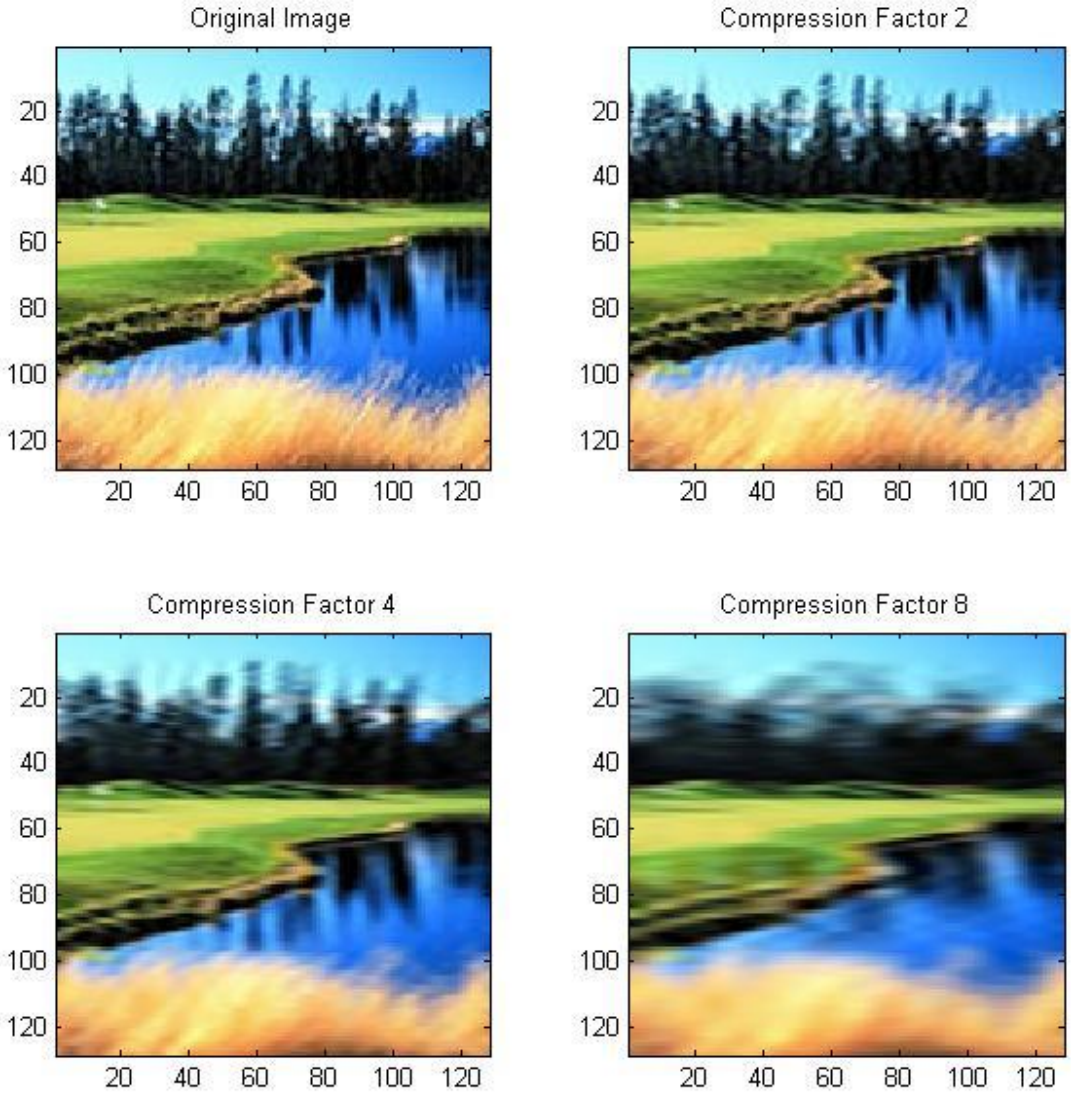
$\mathbf{C_{ij}}$ is the Quantized matrix

On matrix R then IDCT is applied and after that the elements of the new obtained matrix are rounded to their nearest integers. It results in decompressed 8*8 image blocks.

$\mathbf{IDCT\ (R) = Round\ (T^T * R * T)}$

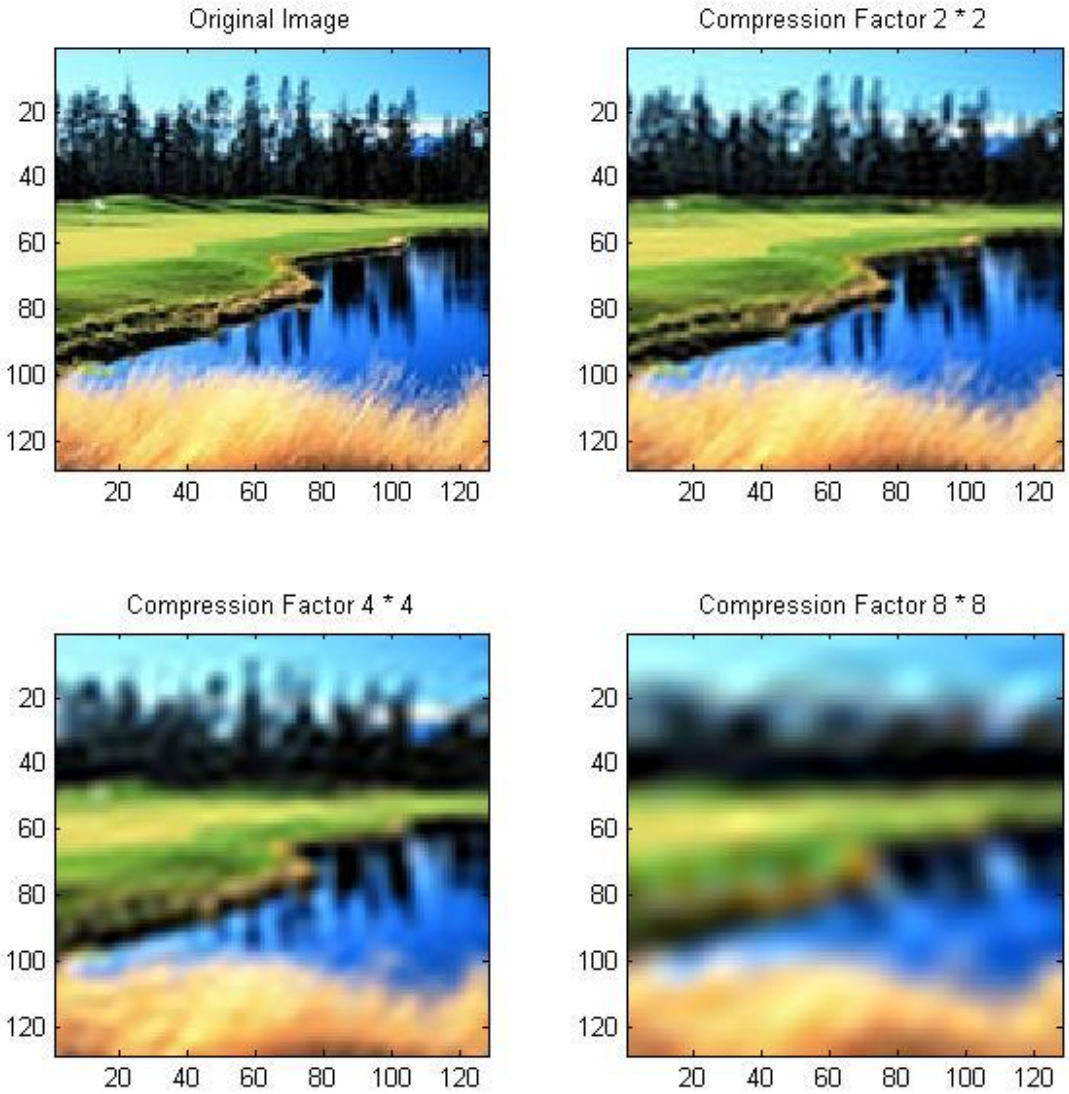
3.1.1.1 Results:

A MATLAB program was written to show the reconstruction via DCT coefficients without using quantization, DCT transform co-efficients are simply discarded with the help of array properties and then the image is re-constructed using IDCT function. In the first part, only rows are transformed using DCT function and row wise compression is only performed. But in the second part column wise DCT of all the three de-compressed images are performed and then those are compressed column wise. After getting the 2D DCT transformed matrix, IDCT is applied and original like image is re-constructed. Results are shown on the next page.



1 2 PSNR2 = 30.3522, PSNR3 = 24.2521, PSNR4 = 20.6470
3 4

In these above images only row wise DCT has been performed and then de-compressed using IDCT. Before decompression some elements have been discarded. First one is the original image, in second, third, and fourth 50%, 75%, 87.5% elements have been discarded from each row before reconstruction of image (before performing IDCT).



1 2
3 4

PSNR2 = 26.6645
PSNR3 = 21.6620
PSNR4 = 18.5925

In these above images both row wise and column wise DCT have been performed and then de-compressed using IDCT. Before decompression some elements have been discarded. First one is the original image, in second, third, and fourth 50%, 75%, 87.5% elements have been discarded from each row as well as column before reconstruction of image.

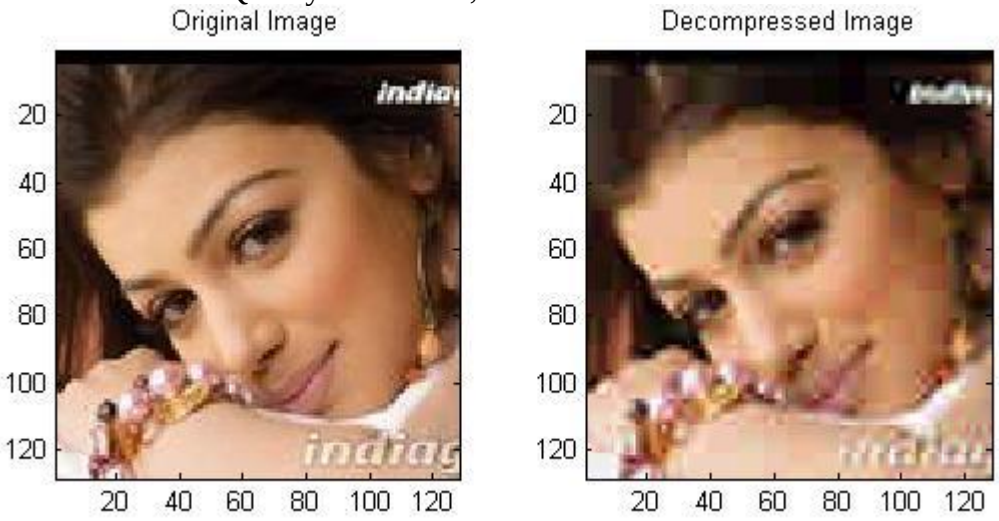
3.1.2.1 Results:

Another MATLAB program was written to show the reconstruction via DCT coefficients, but this time using quantization. Results are shown below.

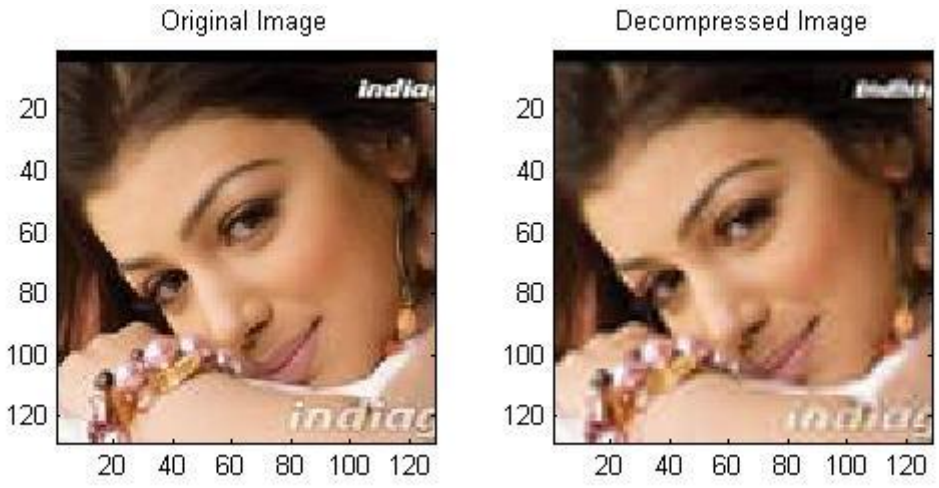
No. of discarded elements = 4(from both rows and columns)

That means 75%elements have been discarded from each 8*8 block before decompression.

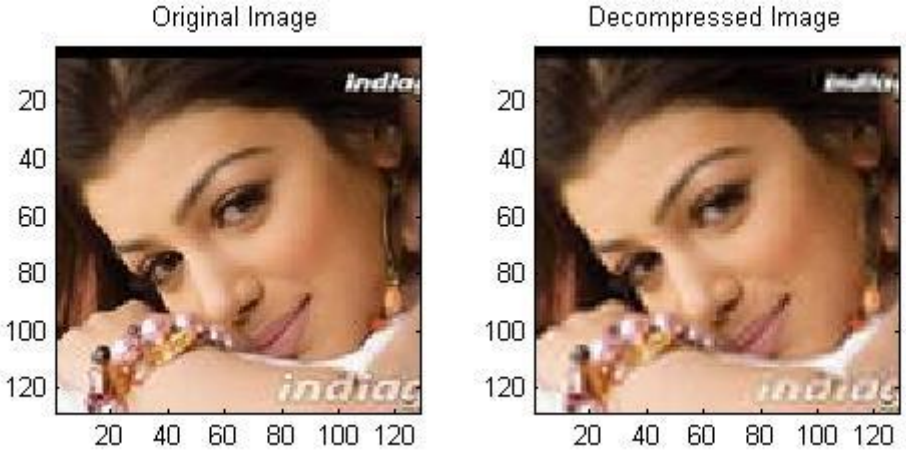
Quality factor = 10, PSNR = 25.9163



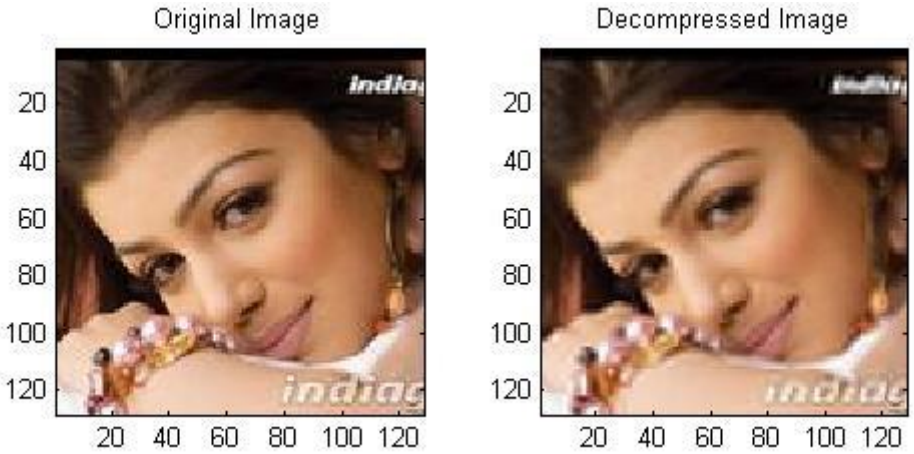
Quality factor = 25, PSNR = 27.5499



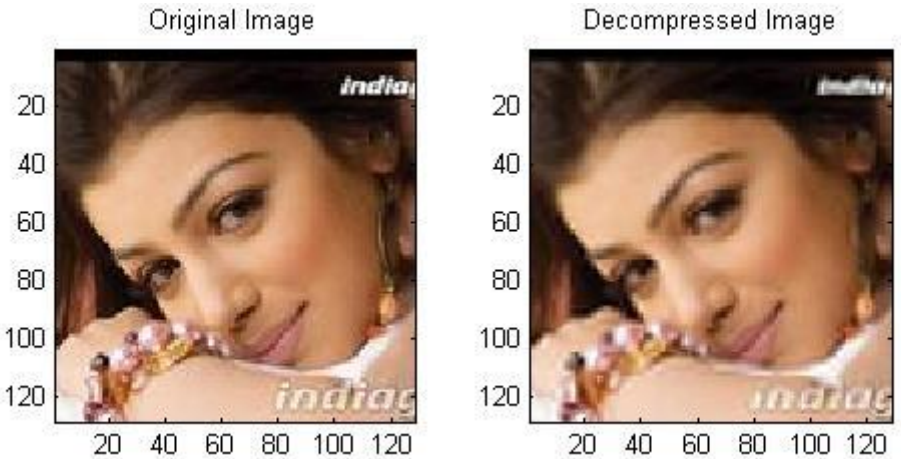
Quality factor = 50, PSNR = 28.0477



Quality factor = 75, PSNR = 28.3725



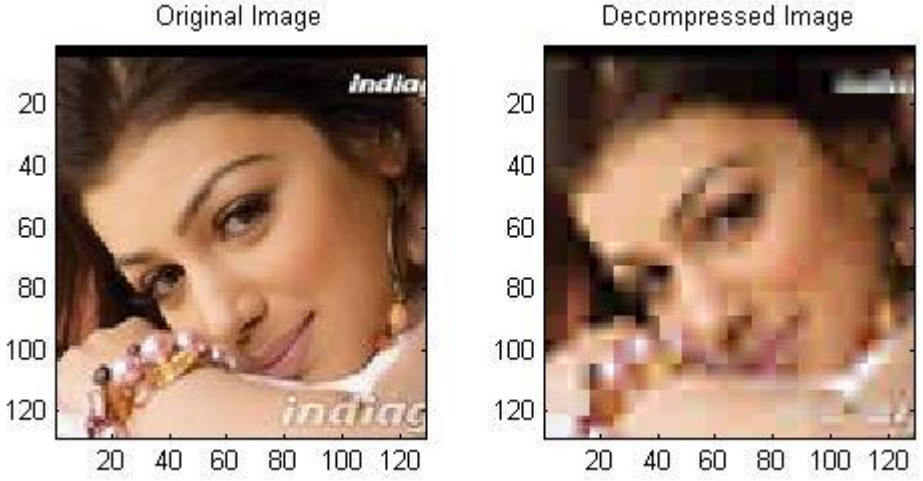
Quality factor = 90, PSNR = 28.3863



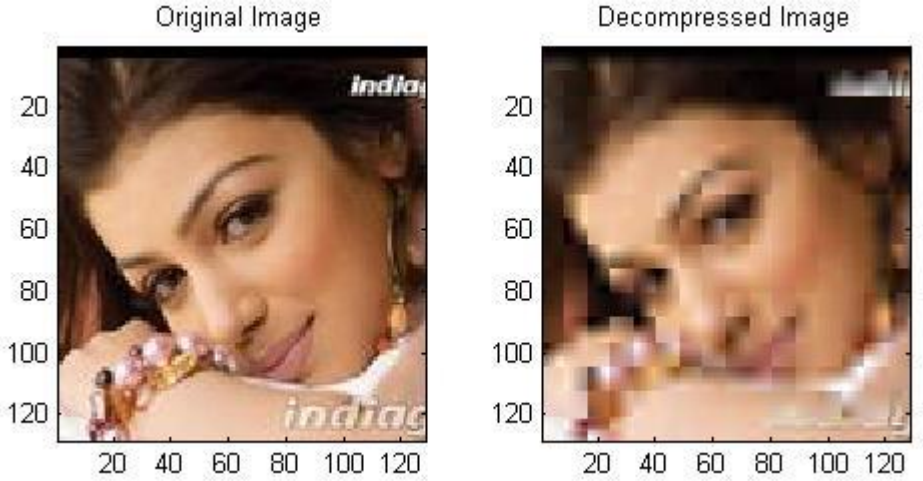
No. of discarded elements = 6(from both rows and columns)

That means 93.75% elements have been discarded from each 8*8 block before decompression.

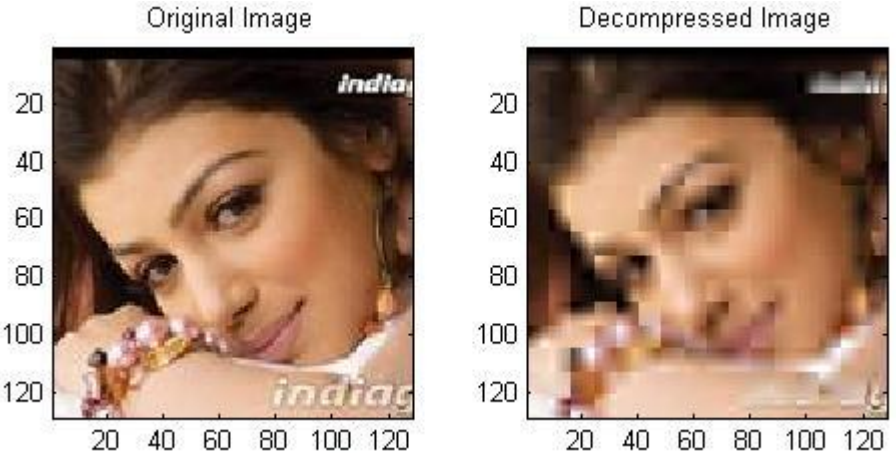
Quality factor = 10, PSNR = 23.1968



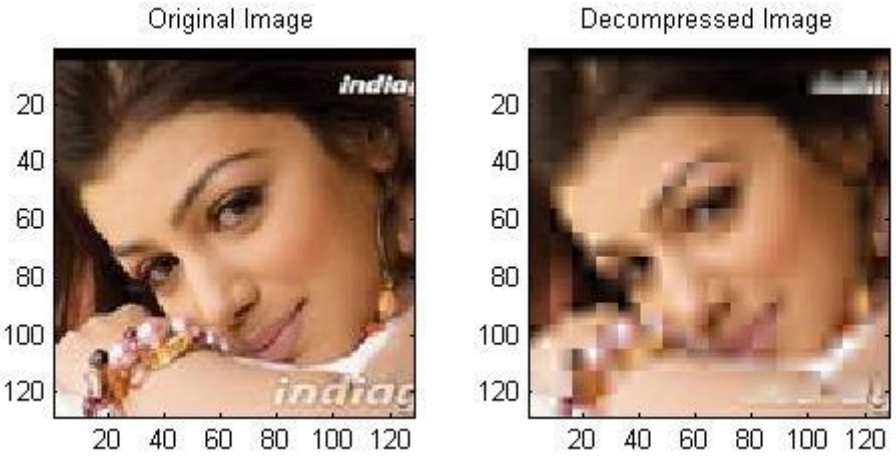
Quality factor = 25, PSNR = 23.5059



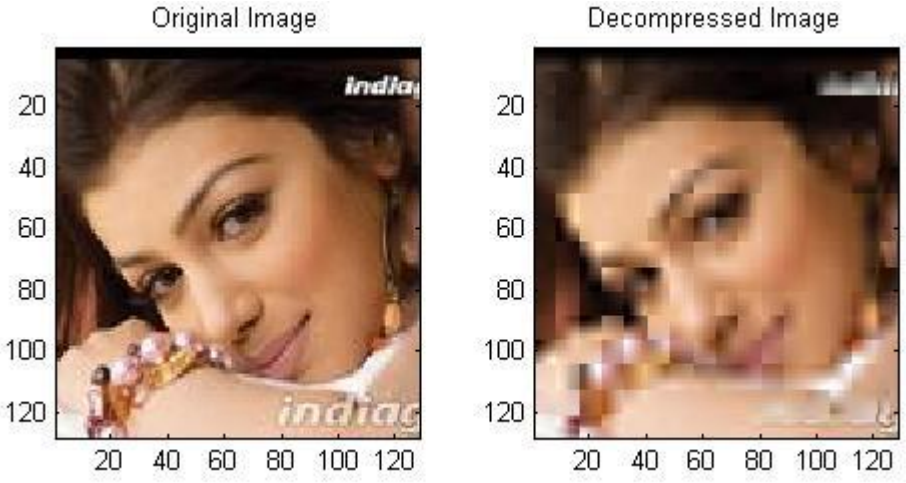
Quality factor = 50, PSNR = 23.5662



Quality factor = 75, PSNR = 23.6066



Quality factor = 90, PSNR = 23.6089



4.1 Discussion and Conclusion:

In JPEG, coding is a very important step of compression which is done by encoding quantized co-efficients in a zigzag sequence using Huffman coding or Arithmetic coding technique. But here coding part has not been done; instead some co-efficients have been

simply discarded from the quantized matrix leaving only few elements. This few elements containing file/data is the actual compressed file. With the help of inverse transforms, original-like image can be reconstructed from the compressed file.

As some elements are discarded to get the compression, the decompressed image is likely to lose some of its information and an image which is approximately equal to the original image is obtained. That's why it's called a Lossy compression. The more elements we discard, the more distorted image we get. Thus it's advisable to remove those many details (elements) which cannot be differentiated by the human eye, which will make it visually lossless.

SECTION II

1.1 Introduction:

Huffman coding can generate optimum redundancy code words in comparison to other present algorithms. In many image and video compression formats like JPEG, MPEG and H.263 etc. Huffman coding technique is used. The Huffman coding is a one type of variable length coding technique; it assigns code words to the symbols as per their probability of occurrence in an image. Most frequently occurring grayvalue gets shortest codeword and the least frequently occurring grayvalue gets longest code word.

This section of the paper aims at the analysis of compressing images by minimizing the number of bits per pixel needed to represent them and also to minimize the transmission bandwidth for transmission of images and then retrieving it back by decoding the encoded data according to a new variable length coding technique which is inspired by Huffman coding technique.

1.2.1 Principle behind Compression:

In every image there is a certain characteristic, it is the correlation between neighbouring pixels which results in redundant information. So therefore the main task here is to find a representation of the image which is less correlated. There are two basic elements of compression such as redundant data reduction and irrelevant data reduction.

a. Redundant data reduction means removal of duplicated data from the original image.

b. Irrelevant reduction of data removes some parts of the image which will go unnoticed by the Human Visual System.

2. Various types of Redundancy:

To compress image file size, there are mainly three types of redundancies which can be removed.

- Redundancy in Coding
- Inter pixel redundancy
- Psycho visual redundancy

2.1 Redundancy in Coding:

A gray level image having a total of n number of pixels is considered. Let the maximum number of gray levels in the given source image is denoted by L (that means the gray levels may have the values from 0 to L-1). Let the gray level r_k occurs at n_k number of pixels. And the probability of occurrence of the gray level r_k is denoted by $P_r(r_k)$. If $l(r_k)$ is the number of bits used to represent the r_k gray value, then average length of each pixel can be calculated from the following formula.

$$L_{avg} = \sum_{K=0}^{L-1} l(r_k)\, P_r(r_k) \qquad (2.1.1)$$

Where,

$P_r(r_k) = n_k / n$

k= 0, 1, 2,……., L-1

Thus n x L_{avg} would give the total number of bits needed to represent the entire image. To get maximum compression ratio L_{avg} has to be minimum (i.e. the length of gray level $l(r_k)$ should me minimum). Thus if coding of the gray levels is done in such a way that the L_{avg} is not minimum, coding redundancy will occur in the image.

2.1.1 Reduction of Coding Redundancy:

For reduction of coding redundancy we can opt for Huffman coding. The Huffman coding is a variable length coding technique which assigns code words to the symbols as per their probability of occurrence in an image. Most frequently occurring grayvalue gets shortest codeword and the least frequently occurring grayvalue gets longest code word.

There are many more techniques to reduce coding redundancy such as Arithmetic coding, Golumb coding, Bit plane coding etc.

2.2 Inter pixel Redundancy

Inter pixel redundancy is related to the correlations between neighbouring pixels of an image. It is because of the reason that the gray value of any pixel of an image can be predicted from the gray value of its neighbouring pixels. Geometric redundancy, Spatial redundancy and inter frame redundancy are all subsets of Inter pixel redundancy. For reduction of the inter pixel redundancies in an image, the two dimensional pixel block is transformed into a more efficient non-visual format. For example, an image can be represented by the difference between the adjacent pixels. This type of transformation is called *mapping*. If original source image can be completely recovered from the transformed data then it can be called *reversible.*

2.2.1 Reduction of Inter pixel Redundancy:

For the reduction of inter pixel redundancies there are various techniques like:

- Predictive coding.
- Delta compression.
- Run length coding.
- Constant area coding.

2.3 Psycho visual Redundancy:

Human Visual system does not analyse each and every pixel or luminance value of the pixel value quantitatively in the given image. It looks for some distinguishing attributes like edges and regions of high textures and tries to combine them mentally to form groupings which can be recognized. These groupings are then correlated with previous knowledge by the brain for the completion of the process of image interpretation. Thus all visual information are not sensed with same intensity by the human eye. Some information has less importance than other information in the visual process. This is called psycho visually redundant information. It can be removed without much degrading the quality of image. Psycho visual redundancy is different from both the coding redundancy and inter pixel redundancy as it deals with the perception of human viewing. Elimination of psycho visual redundant data is a lossy process, thus it is an *irreversible process.*

2.3.1 Reduction of Psycho visual Redundancy:

For the reduction of psycho visual redundancy Quantizer can be used. Elimination of psycho visually redundancy results in some loss of redundant data and that is why it is referred as quantization. As in this process some visual information is lost, it is called lossy compression technique.

3. Types of Compression:

There are basically two categories of compression, one is Lossless compression and another is Lossy compression. In lossless compression techniques, the reconstructed image obtained from the compressed file is numerically equal to the original image. That means there is no difference between the reconstructed image and the original image.

In lossy compression techniques, the reconstructed image is degraded due to loss of information. Generally lossy compression techniques give better compression ratios than lossless techniques. Followings are the different types of lossless and lossy data compression schemes:

3.1 Lossless coding techniques:

- Huffman coding
- Arithmetic coding
- Run length coding
- Area coding
- Entropy coding

3.2 Lossy coding techniques:

1. Transform coding (DFT/DCT/Wavelet transform)
2. Predictive coding

4.1 Explanation of Huffman coding:

The two rules for the Huffman coding are:

a.Shorter code words will be assigned to more frequently occurring symbols and longer code words for less frequently occurring symbols.

b. And same length of code word will be given to the two symbols which occur least frequently.

The Huffman code is obtained by merging the two lowest probability bearing symbols of an image and then again merging two lowest probabilities bearing symbol and this process is continued until and unless only two probabilities bearing compound symbols remain. Hence a

code tree can be generated and by labelling the codes of the code tree, Huffman codes are formed. Following example will explain the whole algorithm:

Table I: Coding table

Symbols	**Probability(Stage1)**	**Code**	**Stage2**	**Stage3**	**Stage4**	**Stage5**
B4	0.3	00	0.3 (00)	0.3 (00)	0.4 (1)	0.6 (0)
B3	0.2	10	0.2 (10)	0.3 (01)	0.3 (00)	0.4 (1)
B5	0.15	010	0.2 (11)	0.2 (10)	0.3 (01)	
B2	0.15	011	0.15 (010)	0.2 (11)		
B6	0.1	110	0.15 (011)			
B1	0.1	111				

In the first column of table I source symbols are sorted as per their probabilities from high to low value. The least two probabilities 0.1 and 0.1 are merged to give another compound symbol of 0.2 probability. Then the new compound symbol probability is placed in the stage-2 column and again the probabilities are sorted from high to low value. This is repeated till only two probabilities remain at the foremost column (0.6 and 0.4 as shown in the above table). The next step is to code each reduced source, starting with the shortest source and following its trail back to its original source symbol. The minimum length binary codes for two source symbols are obviously 0 and 1. As shown in table-I, 0 and 1 binary symbols are attached to the two remaining compound symbols on the rightmost column (attachment can be arbitrary; reversing the order of the 0 and 1 would also work well). Since reduced source symbol bearing 0.6 probability is generated by combining two symbols from the stage-4 column, the binary 0 is assigned to both of the symbols, and a 0 and 1 are arbitrarily

appended to both so that they can be distinguished from each other. Likewise it is repeated for each and every reduced source until the original source symbol is reached. The final code is listed in the third column of the table-I. The average length of the code is calculated as: L_{avg} = (0.3)*(2) + (0.2)*(2) + (0.15)*(3) + (0.15)*(3) + (0.1)*(3) + (0.1)*(3) = 2.5 bits/symbol. The entropy of the source symbols is found out to be 2.41 bits/symbol, which results in efficiency of 2.41/2.5 = 0.964 for Huffman coding technique.

$$\textbf{Entropy(H)} = \sum_{K=0}^{L-1} P_r(r_k) * \log_2(1/ P_r(r_k))$$

Huffman's technique can give optimal codes for a set of given symbols.

4.2 Decoding of Huffman Code:

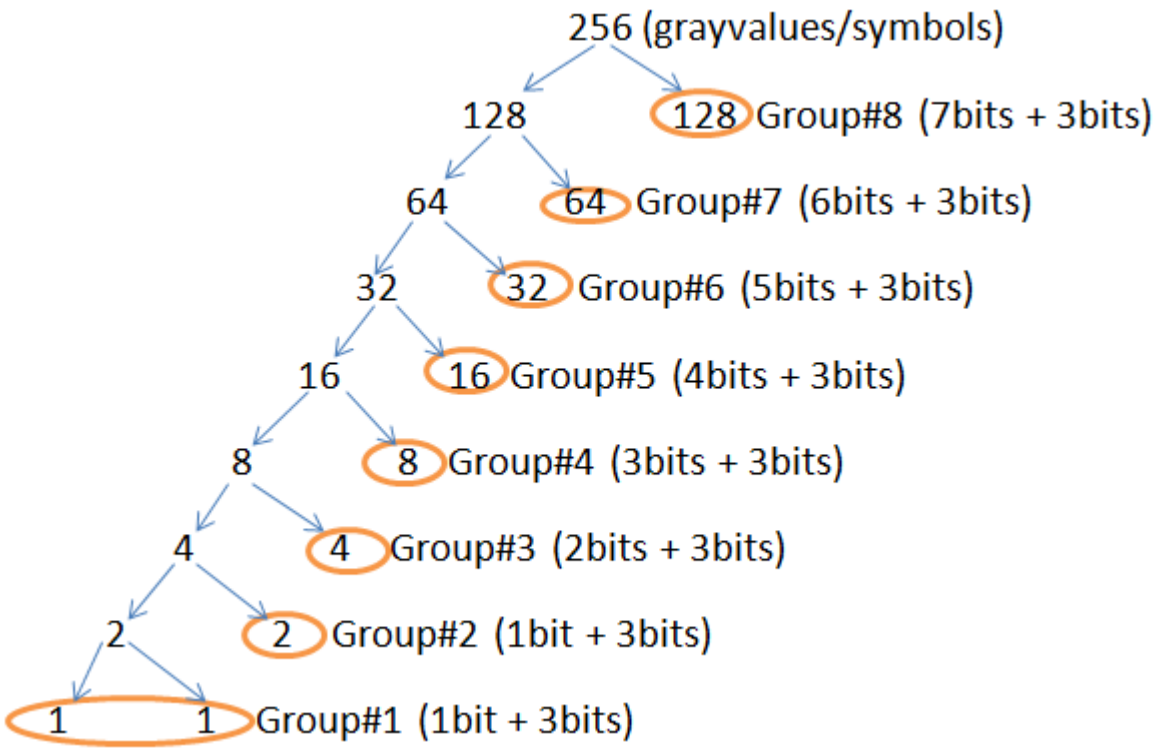
Once the encoded bit stream is formed, decoding can be accomplished using a look-up table. The code is called as a block code since it is uniquely decodable. A fixed pattern of code symbols are assigned to each of the source symbol. Because of the reason that in one way only the string of code symbols can be decoded, codes are uniquely decodable. Hence, any string of Huffman code can be decoded by observing the individual symbols of the binary string from left to right. For example, as per the binary code of table-I, if the encoded string 01001111010 is scanned from left to right, it will give the first code word as 010 (code for symbol B5). The next valid code word would be 011 (code for symbol B2). Code word for the symbol B6 is 110 and for the symbols B3 is 10, continuing in this manner the complete decoded message can be obtained as B5, B2, B6, and B3. So in this way the original image or source data can be decompressed using Huffman decoding.

5. Proposed Technique:

In general images now days that we use are of 8bit graylevel resolution (color depth), 8bit resolution images are only considered.

5.1 Encoding:

8bit graylevel resolution means there are 256 grayvalues. These grayvalues' probabilities (of a particular image) are calculated and are divided into 8 groups as per their probabilities. Groups are divided in such a way that 128 lowest probability bearing grayvalues are in Group #8, next 64 lowest probability bearing grayvalues are in group #7, next 32 lowest probability bearing grayvalues are in group #6 and so on.(See the below figure for better clarification.)



To represent the group no. where the grayvalue belongs to, 3bits(8 groups) are required. Moreover, to represent the grayvalues in those 8 groups, some more additional bits are

required as per the total no. of grayvalues present in a particular group. For example, group#8 has 128 grayvalues or symbols, so to represent 128 symbols 7bits are required. So in total, symbols in the group#8 can be represented in (3+7)= 10 bits/symbol. Similarly, group#7 has 64 grayvalues, so 6bits are required to represent them all. (3+6)= 9bits/symbol are required for group#7 symbols. Likewise, group#6, group#5, group#4, group#3, group#2, group#1 symbols can be represented using 8,7,6,5,4,4 bits/symbol.(Note- Group#1 and group#2 have 2symbols each, and to represent 2symbols 1bit is required.)

In this way, the average length of a symbol out of 256 symbols stays in between 4-10bits/symbol.

Following table II gives an idea about the code pattern to be assigned to each symbol of each group.

Table II: Code pattern of symbols

Group #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Bits/symbol	4	4	5	6	7	8	9	10
Code	000X	001X	010XX	011XXX	100XXXX	101XXXXX	110XXXXXX	111XXXXXXX

The lowest probability bearing grayvalue has ‘1’ in all its ‘X’ bits and the highest probability bearing grayvalue has ‘0’ in all its ‘X’bits. When probabilities of two or more symbols are found to be same they are sorted as per their grayvalue. For example, in a group suppose 0 to 10 grayvalues have same probability, then ‘0’ symbol will have lowest priority and will be given highest binary coded value among those 11 grayvalues and ‘10’ symbol will have highest priority and will be assigned lowest binary coded value among the 11 symbols. In this way the grayvalues are sorted and coded in binary form as per their probabilities.

Binary coded grayvalues of the image are then stored in a file for further use like transmission or storage.

Resulted image file can be found to be smaller than the original raw image file.

5.2 Decoding:

Decoding part is really simple. It's known that first 3bit represents the group number, and from that group it can be known how many more bits should be taken into consideration for the calculation of the actual symbol represented by the code. So from the first 3bits of the bitstream, group number is calculated, then how many more bits are required to represent is calculated and by reverse mapping the symbol is obtained. To get the next symbol same procedure is repeated, starting at calculating the group number from next 3bits of the bitstream. This way each pixel's grayvalue is extracted from the binary coded bitstream, original image is obtained.

6.Results:

Above presented technique is implemented in MATLAB. Results obtained from the proposed technique is compared with the results of Huffman algorithm(results obtained from the inbuilt functions huffmanenco(), huffmandeco(), huffmandict() in MATLAB). It is shown in the following table.

Images	Avg. length of 256 symbols(Huff)	Avg. length of 256 symbols(New)	Avg. length of symbols used in the image(Huff)	Avg. length of symbols used in the image(New)	File size (Huff)	File Size(New)
Cameraman.tif (raw-65536bytes)	9.7344	9.0156	7.0448	7.3174	57,975 bytes	60,207 bytes
Coins.png (raw-73800bytes)	11.3477	9.0156	6.3534	6.4681	58,891 bytes	59,931 bytes
Circuit.tif (raw-76160bytes)	33.7969	9.0156	6.9766	7.5079	66,680 bytes	71,738 bytes

$C_{cameraman} = 1.1304$ (Huffman), $C_{cameraman} = 1.088$ (New)

$C_{coins} = 1.2531$ (Huffman), $C_{coins} = 1.2314$ (New)

$C_{circuit} = 1.1421$ (Huffman), $C_{circuit} = 1.0616$ (New)

7. Observation:

- In case of the new variable length coding technique, due to the grouping system, average length of each symbol among those 256 symbols is constant i.e. 9.0156 bits/symbol. Whereas in Huffman coding technique average length of 256 symbols is variable and from the above results it's found to be greater than the average of the new coding technique, i.e. 9.0156bits/symbol.
- However, average length of symbols which are used in the image (obtained from the new technique) is little larger than the result of Huffman technique.
- Compression ratio of images obtained by the new technique is found to be less as compared to the compression ratio obtained from the Huffman technique.

8. Conclusion:

If the computer programming is done suitably and with less looping implementations for the coding and decoding part of the new technique, it may even take less execution time than the Huffman coding, but with little less compression ratio.

The new variable length coding technique proposed above is found to be falling behind the Huffman technique in almost every aspect, but with a little margin. So it can be used as an alternative to the Huffman coding technique, as the logic behind the new technique is much simpler and is also having less complexity compared to Huffman technique.

REFERENCES

[1] Gonzalez RC, Woods P. Digital Image processing. 3rd ed. New Jersey: Pearson Education 2008

[2] Nelson M. The Data Compression Book. 2nd ed. New York: M&T Books 1995.

[3] Huffman D. *A method for construction of minimum redundancy codes*. Proceedings IRE 1952; 40: 1098- 1101.

[4] Khalid S. Introduction to Data Compression. 2nd ed. New York: Elsevier 2005.

[5] Goloumb G. Run length encoding. IEEE Trans Inf Theory 1966; 12: 399-401.

[6] Rao KR, Yip P. Discrete Cosine Transforms - Algorithms, Advantages, Application. Boston: Academic Press 1990.

[7] Jain AK. Fundamental of digital image processing. India: Princeton Hall of India 1997.